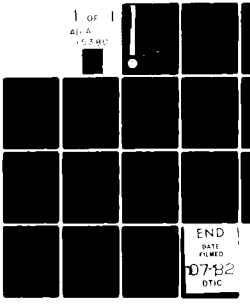
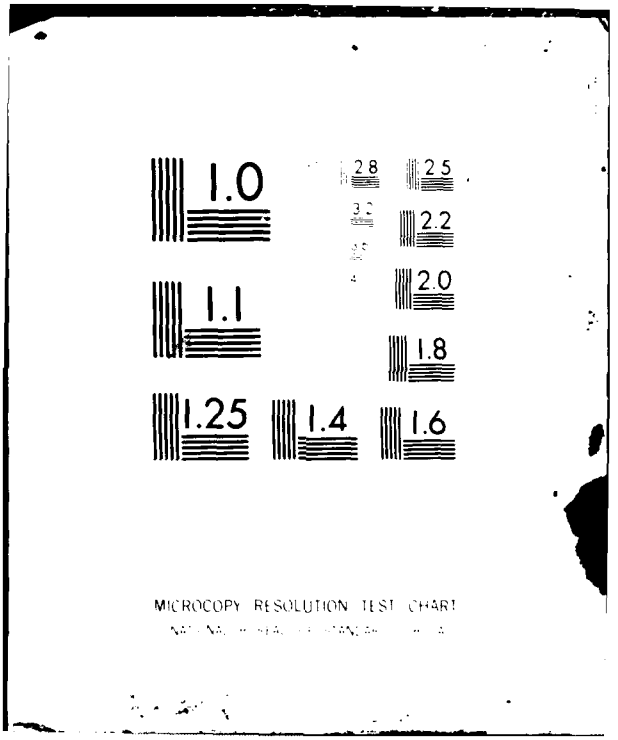
AD-A115 380 STATE UNIV OF NEW YORK AT BUFFALO AMHERST GROUP FOR --ETC F/G 12/1
A MULTI-STRATEGY GAMING ENVIRONMENT.(U)
MAR 82 N V FINDLER, G L SICHERMAN, B MCCALL AFOSR-81-0220
UNCLASSIFIED GCSS-TR-9 AFOSR-TR-82-0465 NL




1.0
2.8
2.5
3.2
2.2
2.0
1.1
1.8
1.25
1.4
1.6
MICROCOPY RESOLUTION TEST CHART

AFOSR-TR- 82-0465

(3)

A MULTI-STRATEGY GAMING ENVIRONMENT

BY

NICHOLAS V. FINDLER, GEORGE L. SICHERMAN,
AND BEDE MCCALL

MARCH 1982

DEPARTMENTAL TECHNICAL REPORT NUMBER: #196

GROUP FOR COMPUTER STUDIES OF STRATEGIES
TECHNICAL REPORT NUMBER: # 9

*THE WORK DESCRIBED HAS BEEN SUPPORTED BY THE AIR FORCE OFFICE OF SCIENTIFIC RESEARCH GRANT [redacted] AFOSR-81-0220

**TO APPEAR IN MAX BREMER (ED.):
GAME-PLAYING PROGRAMS: THEORY AND PRACTICE
(ELLIS HORWOOD: CHICHESTER, ENGLAND, 1982)

STATE UNIVERSITY OF NEW YORK AT BUFFALO

Department of Computer Science

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 82-0465	2 GOVT ACCESSION NO AD-A115 380	3 RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A MULTI-STRATEGY GAMING ENVIRONMENT		5. TYPE OF REPORT & PERIOD COVERED INTERIM, 1 JUL 81-30 MAR 82 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Nicholas V. Findler, George L. Sicherman, and Bede McCall		8. CONTRACT OR GRANT NUMBER(s) AFOSR-81-0220
9. PERFORMING ORGANIZATION NAME AND ADDRESS Group for Computer Studies of Strategies, Department of Computer Science, State University of New York at Buffalo, 4226 Ridge Lea Rd, Amherst NY 14226		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS PE61102F; 2304/A2
11. CONTROLLING OFFICE NAME AND ADDRESS Directorate of Mathematical & Information Sciences Air Force Office of Scientific Research Bolling AFB DC 20332		12. REPORT DATE MAR 82 13 NUMBER OF PAGES 39
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Man-machine systems, automatic analysis and synthesis of strategies, static and learning strategies, generalized production rules, the quasi-optimizer system, the advice taker/inquirer system.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The authors, together with a large and varying number of collaborators, worked on a long-term project aimed at how decisions are and should be made under uncertainty and risk. Learning programs of different types have come to the center of their attention, both in the course of trying to simulate human cognitive behavior and in constructing wholly machine intelligence-oriented competitive strategies. They describe an interactive environment in which an arbitrary number of human and machine strategies, up to a total of eight, can be pitted against each other. The game of draw poker was selected as the vehicle of these
(CONTINUED)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ITEM #2C, CONTINUED: studies since it shares many characteristics with real-life decision-making tasks.

This project has come to an end. The authors also describe their current work on automatic analysis and synthesis of strategies. Although the systems being developed are highly context-independent, they are able to utilize their experience gained in the previous project.

A MULTI-STRATEGY GAMING ENVIRONMENT

Nicholas V. Findler, George L. Sicherman, and Bede McCall

Group for Computer Studies of Strategies

Department of Computer Science

State University of New York at Buffalo

ABSTRACT

We, together with a large and varying number of collaborators, worked on a long-term project aimed at how decisions are and should be made under uncertainty and risk. Learning programs of different types have come to the center of our attention, both in the course of trying to simulate human cognitive behavior and in constructing wholly machine intelligence-oriented competitive strategies. We describe an interactive environment in which an arbitrary number of humans and machine strategies, up to a total of eight, can be pitted against each other. The game of Draw Poker was selected as the vehicle of these studies since it shares many characteristics with real-life decision-making tasks.

This project has come to an end. We also describe our current work on automatic analysis and synthesis of strategies. Although the systems being developed are highly context-independent, we are able to utilize our experience gained in the previous project.

1. INTRODUCTION

Both Artificial Intelligence and its first domain of activity, automatic game-playing, have preceded the advent of computers. Although some of this activity was fake (e.g., von Kempelen's Chess Automaton), some implemented algorithms in "hardware" (e.g., Torres y Quevedo's Rook vs. Rook-and-King mechanical Chess player). Even Turing's and Shannon's celebrated Chess programs were not run on an actual computer.

It is outside the scope of this paper to discuss the reasons and motivations for game-playing programs. We feel that, apart from the intellectual challenge, the by-products of such research justify the effort and resources invested.

In this chapter, we describe a long-term project aimed at studying decision-making under uncertainty and risk, and machine learning. We have used the game of Draw Poker as the vehicle of our investigations [1-9]. As a conceptual and, to a fairly large degree, technical continuation of these efforts, we have engaged in more general, context-independent studies on automatic analysis and synthesis of strategies [10-20].

We first provide an updated version of two surveys of our work published several years ago [5,6]. It is followed by a brief description of our current activity.

2. ON DRAW POKER AND ITS DECISION PROCESSES

The vehicle for studying decision-making must be realistically complex, whether one wants to simulate -- and not caricature -- human cognitive processes or to produce intelligent systems whose performance may surpass that of man. In contrast with one-person vs. Nature or two-person confrontations, a multi-person game allows several strategies to be compared with one another or with some appropriate baseline measure. Futhermore, several human players (situated possibly in different rooms) can compete with machine strategies. Such a laboratory environment

enables one to hypothesize and verify theories of human decision-making, problem-solving and learning processes inexpensively. It also allows Turing tests of a sort to be conducted in which the subjects are asked to distinguish human competitors from programs. Figure 1 depicts the graphic display at a moment during the confrontation between a human player and seven machine strategies.

FIGURE 1 ABOUT HERE

These were some of our reasons for choosing to study Draw Poker, a game popular in many countries [21-26]. Its simple rules (summarized in the Appendix) and limited range of actions, coupled with the depth of analytical reasoning required of any meaningful strategy, render the programming effort invested in a large system "cost-effective". Human players construct and continually modify mathematical models of the game and psychological models of the opponents. Eliciting information on these from them serves the whole range of our research objectives, from generating a sufficiency theory of human behavior to establishing wholly machine intelligence-oriented competitive strategies. We believe that games of imperfect information, and those involving both chance and skill, are more useful for certain studies in Artificial Intelligence than games of pure skill and perfect information -- without trying to belittle the intellectual challenge, the depth and breadth of the efforts needed for programming games in the latter category, such as Chess or Go.

Poker shares many important features of decision-making with "real-life" problems. Such are:

(A) In evaluating alternative courses of actions, the player assumes

(i) a likely "state of nature" based on subjective probabilities,

(ii) plausible (not necessarily rational) actions

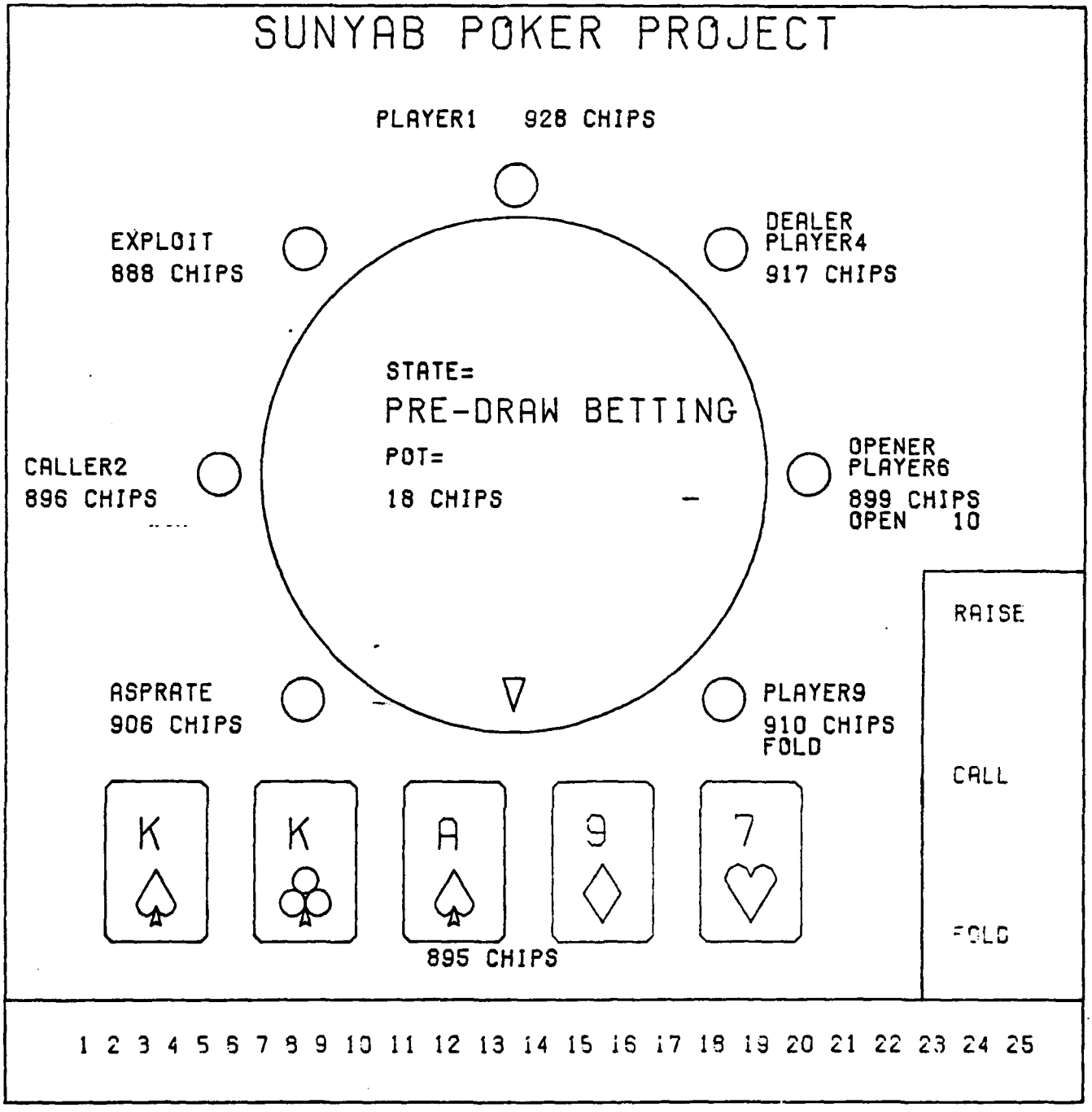


FIGURE 1

A snapshot of the graphics display used as the interactive environment.

by the other participants.

(B) The player can manipulate information by

(i) "buying" information about the others' situation,

(ii) giving away misleading information about his own situation.

(C) Each player has limited financial resources, which he has to manage optimally in the long run. His strategy is the visible projection of his resource management style. His "policy decisions" (concerning, for example, investment for the benefit of projecting a particular style) have obvious analogues in commercial enterprises and political campaigns.

(D) Decisions are made on the basis of probabilities as well as of a dynamic assessment of the competitors, and are guided by tactical and strategical considerations. Tactical considerations refer to momentary and short-term goals (in Poker, for example, within a betting cycle or a game) whereas the strategical ones apply to the whole period of interaction among the same participants (an evening of play).

It is not our aim here to describe how and why Poker, in fact mostly its simplified and abstracted variants, have been used by mathematicians, psychologists, economists and military strategists per se and to illustrate a wide range of phenomena in other domains. Let it suffice to say that Poker with its original rules and with more than two participants cannot be solved by the mathematical theory of games. A realistic analysis of Poker strategies has to employ computer simulation.

3. THE POKER SYSTEM AND THE PLAYING STRATEGIES

The Poker system has evolved over many years and has several times been reprogrammed. It consists of three major modules:

The executive program manages the flow of control and information between the various components of the system. It can also coordinate several interacting jobs. The utility programs collect various statistics, supply public and limited-access information to Player Functions, enable the user to deal prearranged hands as well as those obtained via pseudo-random number generators, help in debugging Player Functions, provide tabulated probability distributions originally derived from Monte Carlo calculations (see below), and direct the so-called tournament mode of play. The tournament mode is designed to eliminate the effects of runs of good or bad hands and of biased seating patterns, similarly to the Duplicate System used in Bridge tournaments. Suppose six players are seated around the table at random and each is dealt a hand at random. Altogether 6!5!=86,400 different hand-player-seat arrangements are possible since one player can be anchored to a constant seat. A tournament consists of a user-specified number of games of different arrangements selected at random without replacement from the 86,400 possible ones.

The Player Functions take the part of the players. They interact with the Executive Program and the Utility Programs by responding to game situations according to the different ordered sets of decision components. A game situation is determined by a player's own hand, and the past and current betting and drawing behavior of all players. The consequences of a game situation, of course, vary among Player Functions. The game situation space can be partitioned according to a structure pattern provided either in advance by the user or, as described later on, by several of the learning programs. The automation of partitioning or, in other words, of the formation of equivalence classes, is fundamentally important. We expand on this issue in our account of the Bayesian players below.

At the outset of the project we ran Monte Carlo calculations to tabulate various distributions of hands, to derive a provisional partitioning of Poker hand classes, and to obtain initial values for certain heuristic parameters for some of the learning strategies.

In the following, we discuss the most interesting player functions out of over 40 strategies we tested. Several of these have not been reported on in [5,6].

3.1. Static Players

The static machine players can be characterized as rigid control structures whose responses depend exclusively upon the game process rather than upon the behavior of the opponents. The decision trees constructed for these players implement heuristics taken from the Poker literature or established by our Monte Carlo runs. There are, however, some additional static players. We single out the RH-player (named after two graduate students, Jean Rachlin and Gary Higgins) and the family of Mathematically Fair Players.

The RH-player follows the principle that a bet should be directly proportional to the tablepot (TABLEPOT); and inversely proportional to the number of people still alive in the game (LIVE), to the number of raises occurred (RAISECOUNT), to the number of opponents still having a chance to say something after him (FOLLOWERS), and to the amount he has to put in the pot to stay in the game (RAISE). Jean Rachlin and Gary Higgins found experimentally the characteristic distribution of a measure of the probability of winning,

$$RH = \frac{TABLEPOT}{LIVE*(RAISECOUNT+1)*FOLLOWERS*RAISE} \qquad (1)$$

and the optimum partitions for Poker hands, in which an approximately constant value of RH calls for a given action in the game.

In order to establish a basis of comparison and a starting point for some of the learning programs, we developed a family of players relying on the Mathematically Fair Strategy (MFS). The fair bet is computed by equating the expected value of winnings to the expected value of losses:

$$p_j \cdot B_o(j,k) = (1-p_j) \cdot [\sum_{m=1}^{k-1} B_a(j,m) + B_f(j,k)] \qquad (2)$$

Here,

p_j is the probability of player j's winning a given hand;

$B_o(j,k)$ is the total contribution of player j's opponents to the pot up to the k-th betting cycle;

$B_a(j,m)$ is player j's bet in the betting cycle m; and

$B_f(j,k)$ is the fair bet to be made by player j in the k-th betting cycle.

As can be seen, this strategy ignores all the second-order effects of the game-situational variables (number of players folding, checking and raising; seat arrangements; nature of opponents and games; etc.) and is obviously incapable of bluffing. This is a serious deficiency against sophisticated human and machine players and the Statistically Fair Player (see below) amends this problem. Bluffing is an essential part of Poker and has a multi-purpose role. First, bluffing has a direct, short-term monetary goal within a single game. By under-representing a strong hand ("sandbagging"), the player tries to keep other players in the game so as to increase the size of the pot. Over-representing a weak hand may result in gain over stronger opponents as well as it can buy information about the other players. Namely, showdowns provide the bluffing player with snapshots of the relationship between the strength of the opponent's hands and their betting behavior. (No showdown takes place if all players but one fold. The bluffing player can force a showdown by paying the price of calling.) Deciding when and

how far to bluff with a given hand and a given history of play against a given set of opponents is one of the key issues in long-term money management. Another major objective of bluffing is to obscure and distort one's strategy and thereby to keep the communication channels noisy.

As said before, Equation (2) leads to a family of the Mathematically Fair Players (MFP's). Many Poker experts recommend, for example, "not to throw good money after bad". That is, a player should consider only the utility of his current investment, an idea which is equivalent to ignoring the summation term in the brackets on the right-hand side of Equation (2). The respective fair bet values are returned by the function FBET (full equation) and FBET2 (summation term omitted). Note that player functions that use FBET2 play more aggressively. Table 1 also contains FBET3 and FBET4. FBET3 returns a bet one chip larger than FBET, and FBET4 lies between FBET and FBET2.

Another distinction among the MFP's is the source of the probability values, p_j. If they come from tables obtained in the Monte Carlo runs (PROB in Table 1), these empirical values grow less accurate as the number of players still in the game diminishes. Certain efficient combinatorial calculations, performed by TROB, a utility routine of our system, provides theoretical probability values during the pre-draw phase of the game. An extension of this calculation also takes into consideration the number of cards drawn by each opponent, and computes the theoretical probability of winning for the post-draw hands (PWIN).

TABLE 1 ABOUT HERE

The 18 MFP's have been tested against a set of standard opponents in runs of 15,000 games each. The runs were made using the Tournament Mode of the system, and players were regeated every 50 games. In each run, two copies of the MFP

Name	Pre-draw functions		Post-draw functions	
	Betting	Probability	Betting	Probability
MFP1	FBET	PROB	FBET	PROB
MFP2	FBET	TPROB	FBET	PROB
MFP3	FBET	TPROB	FBET	PWIN
MFP4	FBET2	TPROB	FBET	PROB
MFP5	FBET	PROB	FBET	PWIN
MFP6	FBET2	PROB	FBET2	PROB
MFP7	FBET2	TPROB	FBET2	PWIN
MFP8	FBET2	PROB	FBET2	PWIN
MFP9	FBET2	TPROB	FBET2	PROB
MFP10	FBET	PROB	FBET2	PROB
MFP11	FBET2	PROB	FBET	PROB
MFP12	FBET	TPROB	FBET2	PWIN
MFP13	FBET2	TPROB	FBET	PWIN
MFP14	FBET	PROB	FBET2	PWIN
MFP15	FBET	TPROB	FBET2	PROB
MFP16	FBET2	PROB	FBET	PWIN
MFP17	FBET3	PROB	FBET3	PROB
MFP18	FBET4	PROB	FBET4	PROB

TABLE 1

The definition of the 18 variants
of the Mathematically Fair Strategy.

being tested participated to reduce further the undesirable "neighbor effects".

The selection of the opponents was based on the speed, memory requirement and stability of the strategies considered rather than their skill and sophistication. We have therefore fixed a set of four static and two learning players for the 18 runs to compare the MFP's. Figure 2 shows for each MFP five measures of quality:

.the final purse size after 15,000 games;

.the average purse size;

.the average win (or loss) per game;

.the "raw win", the percentage of games won by the player;

.the "win ratio", the percentage of showdowns won by the player.

Each of these measures are given as deviations from the respective values averaged over the 18 players so as to make it easier to compare them.

FIGURE 2 ABOUT HERE

We have classified the 18 MFP's into 28 classes according to the betting functions used and the source of the probability values. It would be informative to compare quantitatively the performances of these classes but the figures and tables neccesary for it would make this chapter intolerably long. Suffice it to say that, as we expected, those MFP's that use theoretical probability values, particularly after the draw, outperform those that use empirical ones. Futhermore, an aggressive strategy based on FBET2 before the draw, followed by a more cautious approach based on the FBET after the draw, seems superior to all other choices.

Finally, Figure 3 shows the five measures of quality, as used in Figure 2, for "overall MFP" averaged over the 18 variants and the five standard opponents, each again averaged over all runs.

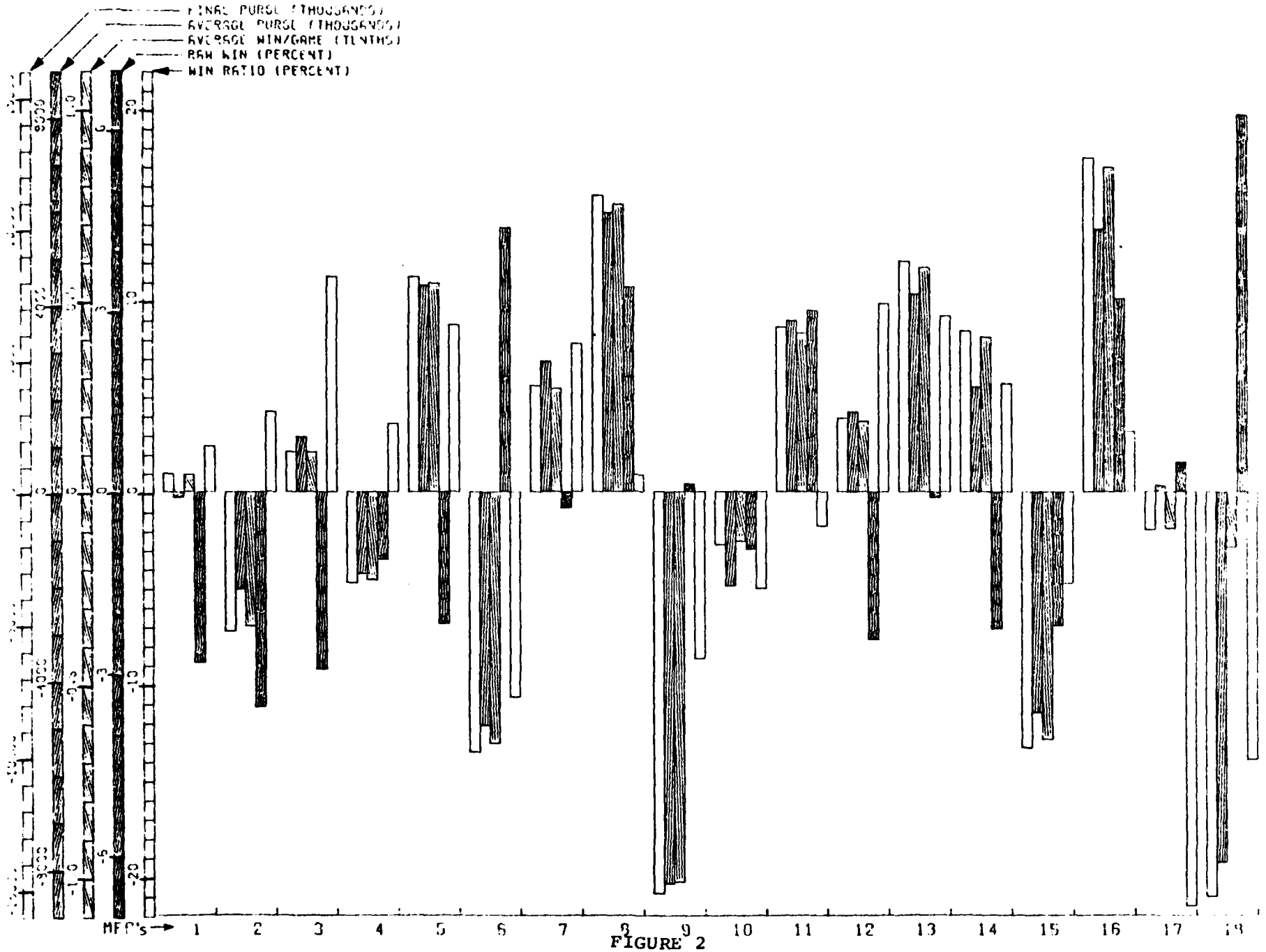


FIGURE 2

A comparison of the performances of the 18 variants of the Mathematically Fair Strategy according to the five criteria (see text).

FIGURE 3 ABOUT HERE

The five opponents were

.the RE-player (alias Player 1);

.three additional static players whose strategies are represented by decision trees: Players 2, 4 and 6;

.the first Bayesian strategy BS1; and

.the Adaptive Aspiration level player (alias ASPRATE).

3.2. Learning Players

Learning any complex activity by humans is, in general, a multistage process involving a number of interdependent factors. Our study of human players was designed to illuminate both the quantitative and qualitative aspects of learning. Several of the human players' qualitative techniques for adapting themselves to the game environment have been incorporated in various machine players. These, however, should not be considered as competitive, independent strategies. Each learning player contains a small number of dynamically changing knowledge components added to some basic set of game rules. We have explored numerous variants of these; namely, how they interact and what influences the rate of improvement in their performance. Besides our interest in various experiments on machine learning, we study these learning processes as models for descriptive theories of human behavior. Also, the Quasi-Optimizer program of our current activity (see Section 4) is designed to generate a normative theory for the Poker environment.

Practically all the learning techniques we have experimented with differ widely from those found in the literature on machine learning. Because of the uncertainty of game actions and their consequences in Poker, we have not used the technique of the evaluation function, which weighs the effects of various characteristic features, or the usual

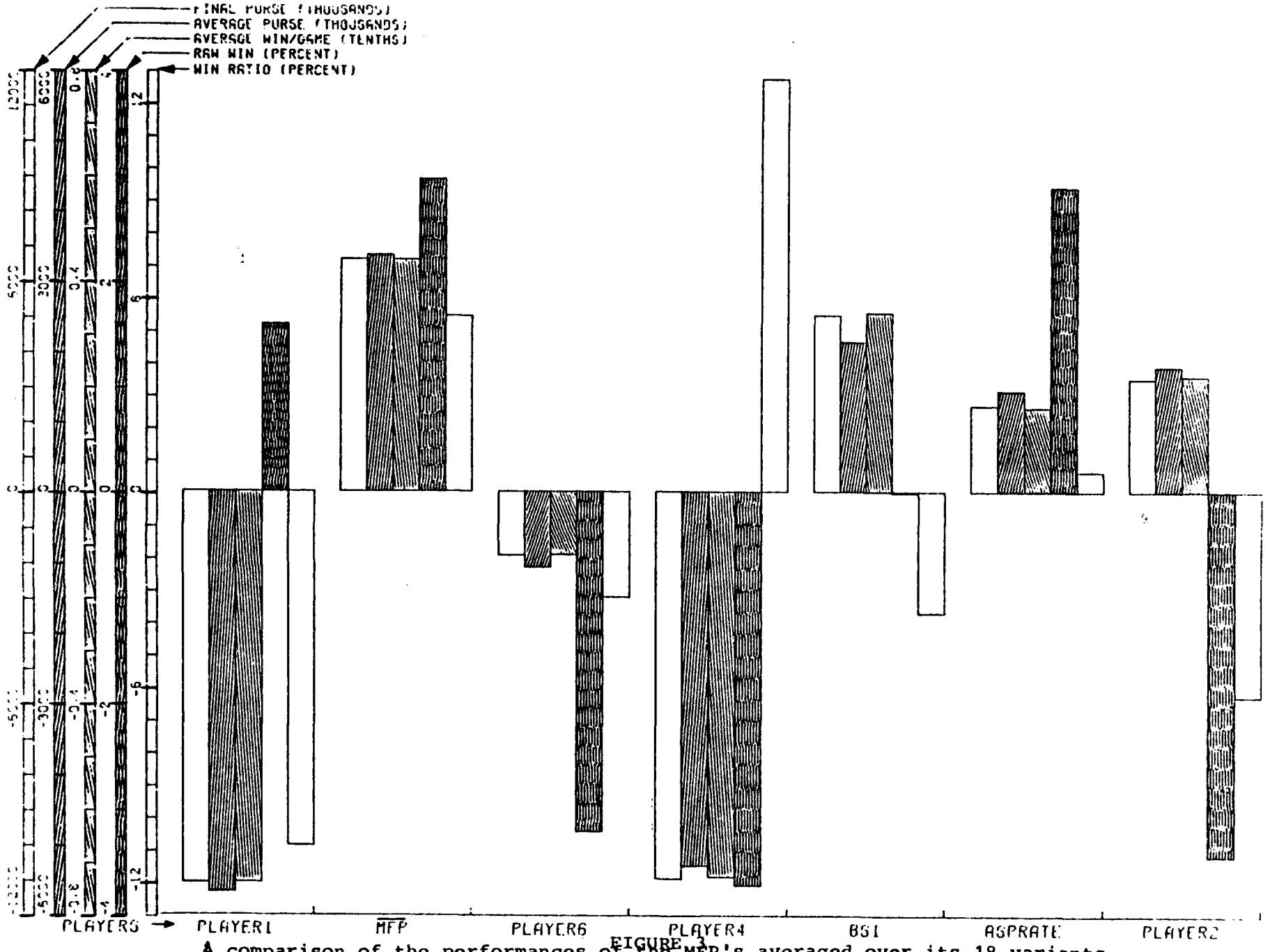


FIGURE 3

A comparison of the performances of the MFP's averaged over its 18 variants, $\overline{\text{MFP}}$, and of the standard opponents, averaged over 18 runs (see text).

minimaxing procedure. In the following, we describe the essential elements of the learning strategies in our project.

3.2.1. The Adaptive Evaluator of the Opponents (AEO)

This player starts with the knowledge necessary to estimate its opponents' hands "roughly"; that is, to come up with a short list of possible cases, on the basis of a few (selectable) indicators. These can be the number and the size of raises in the pre-draw and post-draw betting sequences, and the number of cards drawn -- all these with reference to the opponents' past record. It refines its judgement of each of its opponents as their "personalities" become better known. Every time there is a showdown, AEO updates a statistical data base that correlates the opponents' post-draw betting sequence and their hands.

We have divided the range of all possible hands into 20 partitions, using the principle of equally distributed power of discrimination. Let S(i,j) be some statistic (for example, the ratio of actual bet to fair bet) of player j, collected in partition i; and let m(i,j) and s(i,j) be its mean and standard deviation, respectively. AEO updates these values after each showdown. For reasons explained below, a number of statistics are collected, such as the ratio actual bet/fair bet, tablepot, last bet, total bet, tablepot-last bet, (tablepot-last bet)/tablepot, total bet-last bet, (total bet-last bet)/tablepot, total bet/tablepot, played pot odds/fair odds, and various moving averages of the above.

AEO's initial "rough" estimate of an opponent's hand usually yields a small number of partitions in which the hand is likely to fall. A learning process along three dimensions can reduce this list of possibilities considerably. The first dimension of learning consists of collecting data for the above statistics. Initially, for lack of data, the Mathematically Fair Strategy is assumed and used for estimating hands. Later, but in the early

stage of data collection, the statistical data are pooled over all partitions (of course, using each of the above type distributions separately) and compared with the current value, for possible match. As soon as enough observations are available in every partition, the current value of the statistic, $\hat{S}$, is substituted into the predicate

$$|\hat{S} - m(i,j)| \leq w(i).s(i,j) \qquad (3)$$

for all likely partitions. Here, w(i) is the weighting factor for partition i, initially 0.1 . If (3) holds for one partition, that becomes the estimate of the hand. Otherwise, a learning process along the second dimension takes place. After the showdown, w(i) is (A) reduced by 10% for all partitions incorrectly estimated as "possible", and (B) increased by 10% for the correct but not predicted partition. The weighting factor thus converges to an optimum value in each partition.

We have found that for different opponents different statistical distributions work the best. This is because, although all strategies have the same long-term objective, namely to maximize monetary gain, they use different variables as controllers and indicators in deciding which subgoals to pursue and how to achieve them. The third dimension of the learning process consists of selecting the most effective from among the above statistics.

The combination of the three learning processes produces satisfactory estimates of the opponent's hand. The average, over all players, of the absolute value of the difference between the actual and estimated partitions is less than two -- better for "good" strategies and worse for quasi-random players. AEQ also automatically selects heuristics for evaluation and playing, in a manner analogous to the "Bayesian" approach discussed in Section 3.2.4.

3.2.2. The Adaptive Aspiration Level (AAL) Player

Experimental evidence, described in the psychological literature and also found in our studies with human subjects, indicates the existence of two complementary but not necessarily incompatible attitudes expressed in risk-taking environments. A situation or a sequence of events may "turn on" either a loss-recovery (success-oriented) or a profit-protective (failure-avoidance) mode of operation. (We note here that an individual's behavior may be guided by both needs, that is to achieve success and avoid failure, at the same time. We hypothesize, however, that the attendant anxiety would then disrupt any quasi-rational strategy.) The success-oriented mode of operation induces more aggressive game behavior and (usually) higher bets. The failure-avoidance mode, on the other hand, induces more conservative behavior and (usually) lower bets. However, we considered the change in purse size as a secondary variable whose effects on a player may depend on another, possibly latent variable that is the response to some stimulus configuration. This could be, for example, the difference between the expected and actual gain or loss. The aspiration level represents a cognitive balance between the cost of searching for better outcomes and the value of satisfaction with a safe current status. It is computed by comparing the expected losses incurred while modifying and testing response rules, with the expected long-term gains from improving one's play.

Our implementation was quite flexible in the following way. An "activating mechanism" was established, which can be a function of the change in the financial status, of a significant violation of expectation (losing with a very good hand), or of any experimentally corroborated game-situational variable. The activating mechanism affects the aspiration level, which in turn participates in a two-stage decision process. The latter computes and modifies the mathematically fair bet upwards or downwards, depending on the aspiration level. In another variant of the program, the strategy alters its "rough" estimate of the

opponent's hands obtained by the technique of the AEQ player before its learning process becomes active. AAL then makes its bets according to such altered estimates. In either case, a risk parameter appears in the formalism representing the betting behavior. Different values of this parameter can describe behavior ranging from very timid to extremely wild.

3.2.3 Selling and Buying Players' Images (SBI)

Better players are often willing to invest money in simulating a playing style (conservative, sucker, extravagant, etc.). It is an advertising expense, spent on selling a particular image over a certain number of games. The return on this investment is realized during critical games; the buyers of the image are misled and lose heavily.

With regard to the minimum required length of the selling phase during which stable, observable images are induced, there are some obvious starting points. The farther an image to be sold lies from the mathematically fair strategy, the longer the seller has to present it (more post-draw occurrences). In turn, a conservative player will take longer to buy an image than an adventurous one.

We now introduce a use for bluffing not mentioned previously. Whereas to reproduce a mathematically fair strategy is difficult even for an experienced player, to adhere to some broad playing style is relatively easy. Bluffing, as effected by its frequency and extent, is the most direct means of projecting such a strategy. In other words, the Mathematically Fair Strategy is modified by a probabilistic component determined by the image this player wants to sell. Also, basic betting heuristics can override the fair bet action.

We have explored a number of ways to characterize the opponents. Two dimensions of playing style seem to suffice (cf. Section 3.2.5). The first is the level of consistency, the second the level of cautiousness. The two are not quite independent as our experience with human subjects showed.

For example, a wild player is judged relatively consistent by the others within a much wider range of some indicator variable than a timid player. Or, one would expect a fair player to bet more consistently than an extravagant player would. With reference to the indicator variables, the average win per game and the average loss per game measures cautiousness whereas the scatter of the ratio between the actual and the fair bet reflects the consistency of a player. In another variant, we also tried to quantify the opponents' levels of cautiousness and consistency with the mean value and the standard deviation of the ratio between played pot odds and the fair odds, respectively. The advantage of using these statistics of a single variable as indicators is that we can easily express the relation between the two characteristic dimensions of card playing style.

3.2.4. The Family of "Bayesian" Strategies (BS) That Make Inductive Inferences

In statistical decision theory, Bayes' criterion refers to the choice that minimizes the average expected loss. In our terminology, a "Bayesian" strategy continually readjusts its decision-making rules on the basis of the outcomes of its actions. It collects data on certain characteristics of situations and also on its average gains and losses with various actions in these situations. Finally, this player takes the most profitable action, as suggested by its knowledge base, and updates the respective entries in it. Such a technique should converge to an optimum strategy against non-learning opponents.

Because of limitations in computing time and memory space, only a few relevant features can be extracted from the situations. The first and simplest Bayesian strategy, BS1, observed only its hand and assigned it to one of 11 classes. Moreover, it could take only three "actions", each a strategy for the entire game. Subsequent Bayesian players were extensions of BS1: CALLER2 also observed which opponent

was betting against it; BS3 observed how many rounds of betting had taken place; and SCHNE (named for Bill Schneider, who programmed it) observed the value of the winning hand in each game, and the maximum and average pot with each hand value. All three players need larger tables of results than BS1.

Another problem with Bayesian players is controlling the amount of experimentation. In a given situation a player must try each action at least once. BS1 tries each action once only, which sometimes causes a disorder that we call Bayesian withdrawal. As an example, suppose that the first time BS1 holds a flush, it bets the limit, and is beaten by a full house. This suffices to deter BS1 from ever betting the limit again with a flush, simply because its average gain (over one game!) is negative. In our early Bayesian players, the effect of Bayesian withdrawal was mitigated, though not eliminated, by averaging the past results for a class of hands with the results for neighboring classes. SCHNE first plays a set number of games with a fixed strategy, and thereafter bases its strategy on its previous results.

During the past two years we have refined the Bayesian model with a view to training it for human competition. Our last model, BS8, incorporates several improvements over the old BS3. The most significant change was to observe how many cards the opponent draws instead of how many rounds of betting take place after the draw. We also rewrote the hand classifier to classify hands according to their "D-values" instead of their Poker ranks. The D-value is the probability of a hand's being better than a randomly dealt one.

The consequences of an action in an early round of betting are harder to judge than those in a late round. We therefore made BS8 do more "forced experimentation", i.e. deliberate deviations from recommended actions, early in the game rather than in its later stages. Since even this precaution will not necessarily prevent Bayesian withdrawal,

we instructed BS8 to continue making occasional experiments throughout a session. As a further precaution, we implemented recency weighting, whereby after each game the tables are multiplied by a factor slightly less than unity. This factor is called the oblivion coefficient.

We also eliminated the practice of drawing four cards to a high-card hand. The lore of Poker discourages drawing four cards as a sign of weakness; our tests confirmed this lore.

The great strategic weakness of the Bayesian model is that it cannot learn to bluff. The main purpose of bluffing with a poor hand is to encourage the opponents to bet when you have a good hand. The basic Bayesian model ignores such interactions. We therefore added a new module, the Bluffing Supervisor, to the Poker System. The Bluffing Supervisor maintains statistics on all bluffs during a session and computes how often a player ought to bluff. It turns a naive player function into a sophisticated bluffer.

These improvements caused BS8 to play remarkably like a human player. We conjecture that by observing one more variable--the position of the dealer--BS8's strategy would equal or surpass most human strategies. Of course, this change would multiply the size of BS8's tables by the number of players. Instead, we introduced a more effective measure, the skill of the opponents. By monitoring its opponents' play, BS8 classifies their strength as expert, average, or novice. For each strength, BS8 maintains separate tables, each with different oblivion coefficients.

We are still awaiting the results of extensive human testing, which a private research group is performing. In our own tests with human volunteers, BS8 held its own, eventually showing a profit after converging to a sound strategy.

3.2.5. The Statistically Fair Player

We noted in Section 3.1 that the Mathematically Fair Players do not bluff -- a serious shortcoming against sophisticated opponents. They can, therefore, be outguessed by a strategy that recognizes this fact. The Statistically Fair Player (SFP) implemented by Terence L. Roy [27] eliminates this deficiency. It can also identify MFP or near-MFP opponents and respond to them appropriately.

The SFP analyzes the statistics gathered on the other players over all past games to adjust the frequency and amounts of its bluffs. It attempts to adopt a style similar to those opponents' whose strategy is not near the mathematically fair one -- a common recommendation in the Poker literature -- and to bluff heavily and often against MFP and near-MFP strategies. (Such an approach makes sense because most Poker end games, in the laboratory and in real life, are two-person confrontations.)

The statistics collected are the mean value, m, and the standard deviation, s, of the ratio between the odds played and fair odds of each opponent. (The period over which the data are used is the most recent 300 bets, with the last 100 being weighted double. Thus SFP also adjusts to changes in the opponent's playing style.)

The program maps the relevant values of m and s into measures controlling the frequency and extent of bluffing so that SFP responds to MFP and non-MFP players as described above. This is a rather elegant and inexpensive technique for characterizing the levels of cautiousness and consistency exhibited by the other players.

Finally, we note that when several opponents are still in the game, SFP weighs its response to them according to their current purse size. In other words, richer (and better) strategies are considered more important.

3.2.6. Programming the Zadeh Strategy

A member of our group, C. E. Pearson, implemented [28] our first comprehensive strategy intended for humans as distinguished from the isolated techniques from Poker books

adopted by most of our previous player functions. He decided to approximate a mathematically and psychologically well-founded strategy designed by Zadeh [25].

The cornerstone of Zadeh's strategy (indeed, of any sound strategy) is the observation of what each player opens with. Since the Poker System has no routines for doing this, Pearson had to design and write them. After he modified our "mathematically fair" equation (2) to use information about what hands each player opens with, he obtained tables that agreed almost perfectly with Zadeh's tables of probabilities and recommended actions.

Zadeh distinguishes three eventualities during the betting: (1) all the opponents fold; (2) at least one opponent raises; and (3) at least one calls, but nobody raises. Pearson combined cases (2) and (3) to simplify the algebra. The expected gain from raising is then the sum of the opponents' contributions to the pot, multiplied by each opponent's probability of losing by (i) folding or (ii) being beaten in the showdown. The expected gain from calling can be computed similarly.

Another component of Zadeh's strategy is bluffing. Good players bluff rarely, but they do bluff. The Zadeh player function uses "Reflective Bluffing", a strategy well-suited to a computer. According to the original proposal [29]: "Suppose a player wishes to bluff b percent of the time. If his hand lies in the n-th percentile from the bottom, where $n < b$, then ... he would pretend that his hand is in the n-th percentile from the top." This elegantly implements Zadeh's dictum: "Bluff with your worst hands." It also tends to deter opponents from calling a bluff; when a player who uses it seems to hold a very good hand, there is a 50% chance that he really holds it.

To consider its opponents' bluffing frequencies, the Zadeh function must recognize when an opponent has bluffed. Pearson's implementation simply counts the times an opponent shows a hand too weak to open with. Though it neglects the times an opponent folds or improves after having bluffed, it

appears to be good enough for practical purposes.

The final function won against the static strategies convincingly. We could not obtain enough core memory to test the Zadeh function against other learning strategies.

3.2.7. A Sample Run

Figure 4 shows the change in purse size vs. the game number for six competing strategies over 15,000 games.

The participating players are:

.the Adaptive Evaluator of the Opponents (AEO), PLAYER9;

.the Adaptive Aspiration Level (AAL) player, ASPRATE;

.the strategy Selling and Buying Player's Images (SPI), EXPLOIT;

.the first Bayesian Strategy, BS1;

.the fourth Bayesian Strategy, SCHUE;

.the eighth Bayesian Strategy, BS8.

Some comments are necessary. BS1 learns much faster than BS8 but, asymptotically, it is inferior. (Note that BS1 has reached a plateau after about 14,000 games whereas BS8 monotonically improves after about 9,500 games.) BS1 is "egocentric", perceiving only its own hand as a situation-descriptor. Therefore, the type and number of the opponents and their actions do not matter. In contrast, BS8 does observe the opponents' actions. However, to save memory space (and also because BS8 was originally designed to play against a single human opponent), it does not distinguish between the other strategies. Of course, one has to pay a price for such an over-generalization in BS8's playing quality under such conditions. BS8 does win "hands down" when in its element, two-person games, after a sufficiently long training period.

FIGURE 4 ABOUT HERE

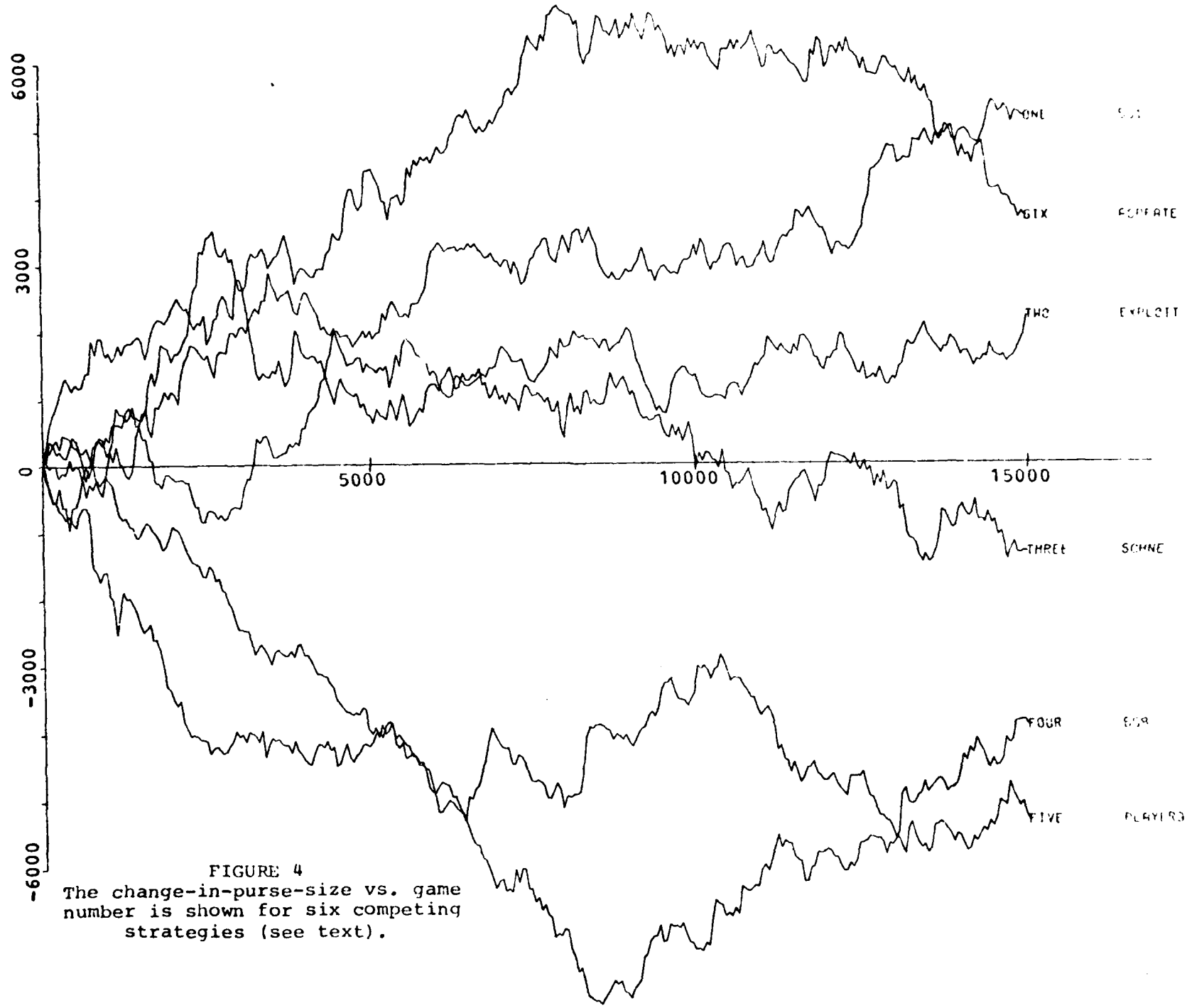


FIGURE 4
The change-in-purse-size vs. game number is shown for six competing strategies (see text).

4. OUR CURRENT ACTIVITY

Relying on the experience gained in the work described above, we are now engaged in a long-term effort aimed at automatic analysis and synthesis of strategies. Our objectives can be summarized as follows:

.to identify adequate computer representations of static and learning strategies, which representations can then be effectively and efficiently employed both in a simulated world and in direct interaction with the real world;

.to develop techniques which analyze strategies, measure their performance, and identify and evaluate their components ("credit assignment"), under most or all conditions;

.to observe strategies in action--either in a sequence of unperturbed confrontations with others or under "laboratory conditions" when the environment is specified according to some experimental design--in order to generate computer models ("descriptive theories") of them;

.to combine the best components of several strategies, eliminate the redundancies and inconsistencies among these components and produce a strategy that is normative in the statistical sense;

.to establish stochastic, causal relationships between open variables that can be measured at any time and hidden variables whose values can be identified only intermittently or periodically, in order to find out the actions of a strategy, and their underlying reasons and consequences;

.to create a system that can be taught strategies via principles and high-level examples, able to make inquiries about vague, incomplete or contradictory advice, and to apply, evaluate and improve the strategy so acquired.

Next we describe the major characteristics of three projects in this area.

4.1. The Quasi-Optimizer (QO) System

Let us consider an environment in which either several organizations are competing to achieve an identical, mutually conflicting goal, or else a set of alternative strategies exist, each trying to win against an identical, opposing strategy [7,11,19]. (One can assume, for the sake of generality, that a goal vector is specified whose components need not be independent in real-life confrontations; for example, in air battle management, the ratio of targets accessed and enemy air defense units suppressed are obviously inter-related goal components.)

Each strategy evaluates the environment by measuring certain variables (numerical or symbolic) available to it, which the strategy considers relevant. Such variables may be the real or assumed actions of the adversary, the perceived state of the confrontation, availability and capabilities of friendly forces, threat estimates, criticality and vulnerability of the adversary's and our resources, etc. An important component of a strategy is interpreting these measurements and incorporating them in the process of making decisions that can lead to goal-achievement (and to prevent goal-achievement by the adversaries).

The environment as perceived by the strategy is unclear because some information may be unavailable, missing (risky or uncertain, according to whether or not the relevant a priori probability distributions are known, respectively) or obscured by noise (caused accidentally or by deliberate obfuscation). If the decisions based on such incomplete or inconsistent information are less sound than those of the adversaries, resources will be wasted and goal achievement will be farther removed.

Let us now consider how we could generate a new strategy. The system has to generate automatically a model (a descriptive theory) of every participating strategy through observation and measurements. It then has to assign to each component of the models some measure of quality; that is, an outcome-dependent allocation of credit must be

made.

The strategy obtainable from the best components of the model strategies is a normative theory which is potentially the best of all available ones, on the basis of the information accessible by us. This normative strategy is in fact only quasi-optimum for four reasons. First, the resulting strategy is optimum only against the original set of strategies considered. Another set may well employ controllers and indicators for decision-making that are superior to any in the "training" set. Second, the strategy is normative only in the statistical sense. Fluctuations in the adverse strategies, whether accidental or deliberate, impair the performance of the QO strategy. Third, the adverse strategies may change over time and some aspects of their dynamic behavior may necessitate a change in the QO strategy. Finally, the generation of both the descriptive theories (models) and of the normative theory (the QO theory) is based on approximate and fallible measurements.

The system under development employs the following modules:

4.1.1 The QO-1 [11] assumes a monotonic strategy response surface and uses either exhaustive search or binary chopping to construct a descriptive theory of static (non-learning) strategies.

4.1.2 The QO-2 [15] extrapolates a finite sequence of learning trees, each representing the same strategy at different stages of development, and computes their asymptotic form. The latter will then be used in constructing the normative theory.

4.1.3 The QO-3 [17] minimizes the total number of experiments QO-1 has to perform. It no longer assumes that the strategy response surface is monotonic and will eventually also deal with multi-dimensional responses. QO-3 starts with a balanced incomplete block design for experiments and computes dynamically the specifications for each subsequent experiment. In other words, the levels of the decision variables in any single experiment and the

length of the sequence of experiments depend on the responses obtained in previous experiments.

4.1.4 The QO-4 performs the credit assignment. That is, it identifies the components of a strategy and assigns to each a quality measure of the 'outcomes'. An outcome need not be only the immediate result of a sequence of actions prescribed by the strategy but can also involve long-range consequences of planned actions.

4.1.5 The QO-5 constructs a 'Super Strategy' by combining strategy components associated with outcomes of a quality above a threshold value.

4.1.6 The QO-6 generates a Quasi-Optimum strategy from the Super Strategy by eliminating its inconsistencies and redundancies. It also tests and verifies the QO strategy for completeness.

4.2. The Advice Taker/Inquirer System (AT/I)

The objective of this system [8] is to establish a man-machine environment in which a human advisor can teach strategies of confrontation on-line, through principles and high-level examples. The principles and examples normally consist of situations and recommended actions. (Principles describe rather general situations defined in a flexible manner whereas examples are specific and illustrate appropriate behavior in a general situation by analogy with a particular one. Actions can either adhere to some general guidelines or follow a set of sharply defined prescriptions.) Whenever the system finds the advice given to be vague, incomplete or inconsistent with previously imparted knowledge, it makes inquiries and asks for clarification. The advisor can define and re-define the components of a principle at any time. He can also override temporarily the strategy taught so far by issuing an order.

The system does not start out with a blank memory. It knows the rules governing the confrontation, the variables, and the ranges of their values within the situation space. The advisor can at any time

(i) define variables, functions, general and specific actions, confrontation-related adjectives, nouns and verbs--in terms of constants, confrontation parameters, current values, overall and moving averages of statistical values, basic confrontation actions, and Boolean and relational operators;

(ii) define principles of a strategy which connect a situation (specified as a Boolean combination of ranges of statistical variables--again current values, overall or moving averages) to some general or specific action;

(iii) give high-level examples by connecting sharply specified situations to direct confrontation actions;

(iv) make inquiries about definitions, principles, and values of statistical variables stored so far;

(v) issue an order which temporarily overrides the strategy acquired so far.

In turn, the system can

(a) ask for clarification whenever new definitions are vague or conflict with stored ones, or the strategy is incomplete in not covering the whole confrontation space;

(b) return exemplary actions in user-specified confrontation situations, in accordance with the strategy acquired;

(c) display definitions, principles, confrontation parameters, values of variables, etc.

Random number generators also have a role in defining game-theoretically mixed strategies. A sense of time has also to be incorporated in the "tool kit" of definitions, whether it refers to real time or to an event counter.

We note two important facilities to be used in specifying principles. Let us call these Advisor-Assigned and Advisor-Defined Adversary Types (AAAT and ADAT, respectively). In the former case, the advisor assigns a certain adversary to one or more categories (Adversary Types) named by him. In the latter case, the advisor defines categories by Boolean combinations of ranges of

statistical variables, which are regularly or continually collected over the adversary's actions. (The variables can refer to current values, or overall or moving averages.) At prescribed intervals, the system compares the adversary behavior with the specifications of all ADAT's. Accordingly, each adversary (at that time) may belong to various Advisor-Defined Adversary Types. Thus the principle can prescribe an action for all such adversaries that satisfy the definition conditions of the Adversary Type at hand.

Advisor-defined nouns can reasonably be required to be unambiguous. However, adjectives (and, to some extent, verbs) must often have different meanings when used to modify different types of nouns (cf. a "strong attack" vs. a "strong concentration"). The AT/I system has to distinguish (at least) four different classes of instances:

(i) Patent: confrontation parameters, statistical variables, AT/I's own resources (e.g., "If your air superiority is more than 2:1, seek air battles.")

(ii) Interactive: the adversary's actions during current confrontation (e.g., "If the adversary is bringing up additional resources, assume a holding position.")

(iii) Statistical: accumulated data about the adversary's past behavior (e.g., "If the adversary is self-confident, make sudden attacks.")

(iv) Inferential: assumptions about the intentions or events behind the adversary's behavior (e.g., "If the enemy appears to have received additional supplies, wait for confirmation.")

This classification is neither exhaustive nor exclusive. If the Definition Manager, a part of the programming system, cannot decide unambiguously how to classify components of the definition, it has to consult the human advisor.

Another difficulty rests with the need to resolve a situation-dependent conflict between principles of global and momentary relevance. Furthermore, the system must be able to generate disambiguating questions whenever the advisor specifies inconsistent priorities for the principles.

Finally, we note that to teach a strategy by telling how to do things in general is more efficient and less error-prone than to tell what to do in every relevant situation. An AT/I-like system would have practical usefulness in doing this. Human experts would specify, via a sophisticated interaction with the machine, a number of alternative strategies. Other components, such as a QO-like system, would then generate uniformly structured models of each strategy. A prescriptive, quasi-optimum strategy would then finally be constructed from these.

The system under construction employs the following modules:

4.2.1 The AT/I-1 constructs the framework for the flow of information and control between the AT/I system and the advisor.

4.2.2 The AT/I-2 converts the principles and high-level examples into a canonical form and stores them. Next it embeds them into an initially skeleton strategy which then becomes employable.

4.2.3 The AT/I-3 eliminates inconsistencies and incompletenesses from the strategy acquired, in part by interacting with the advisor.

4.2.4 The AT/I-4 tests (verifies) and evaluates the strategy constructed according to a metric which is independent of any particular strategy.

4.3. The Generalized Production Rules System (GPR)

The underlying motivations for the actions prescribed by a strategy, the actions themselves, and their consequences are not necessarily observable and measurable at any desired time. The values of such hidden variables can be identified only at certain times, either intermittently or periodically. At other times, their values have to be estimated. In contrast, the open variables are readily measurable at any time. The estimation is based on generalized production rules expressing stochastic, causal relations between open and hidden variables. Either can be cause or effect. The GPR system [10,12,16,18] is designed to provide decision support for expert systems in need for numerical estimates of hidden variable values.

A knowledge base is established over a period of measurements. It consists of an ordered set of generalized production rules of the form

$$W_r/M_{ijk}/T_{jm} \Longrightarrow V_m(H_n) \qquad : Q_r \qquad (4)$$

Here W_r is the number of rules that have been pooled to form the r-th rule. M_{ijk} is the i-th combination of the parameters of the j-th basic pattern (morph) [13] describing the behavior of the k-th open variable (OV). T_{jm} is the difference in time (timelag) or in space (distance) between the start of the j-th morph (in case of a trend) or its occurrence (in case of a sudden change or step function), and the point of time or space at which the n-th HV, H_n, assumes its m-th value, V_m. This difference may be positive--when the OV is the cause and thus precedes the HV, the effect--or negative in the opposite case. The term 'lag' is used for T_{jm}, whether it refers to a timelag or distance. Q_r is the credibility level of the r-th rule. It lies between 0 and 1, and depends on two factors:

.how well the morph fits the datapoints over its domain, and

.how many and how similar the rules were that have been pooled to form the rule at hand.

When an estimate of a HV value is desired at a certain value of the lag variable, the user has to provide in its vicinity a sequence of values of all available OV's that are assumed to be causally related to the HV. These sequences are then submitted to the morph-fitting program (MFP). The system then looks in the knowledge base for the N best estimates (N specified by the user) coming from rules that

.connect the HV sought and the available OV's;

.refer to the same type of morph as the newly fitted one;

.involve morph parameters and lag values that are "similar enough" to those in the query, i.e. that are within the user-specified range of pooling rules.

The so-called confidence level of the estimate, C_e, depends on the credibility level of the rule used as well as how well the new morph fits its datapoints and how close its parameters are to those of the morph matched in the knowledge base.

Let us now assume that the estimation is performed and up to N values of the HV are returned for each lag value that yields such a possibility. The system will calculate the average of the N estimates weighted by their confidence levels. This process thus provides datapoints, each specifying weighted average HV vs. lag value, over the whole range of interest. The system then finally invokes the MFP to produce the functional form desired. Its validity is based on the assumption that the OV's, whose morphs were used for the estimation, obeyed the same laws when the observations were made for the knowledge base as when they were measured for the estimation. Furthermore, the relations between and within the groups of OV's and HV's do not, statistically speaking, vary over time.

The system employs the following modules:

4.3.1 The GPR-1 fits a minimal set of basic patterns, morphs, to a sequence of open variable datapoints.

4.3.2 The GPR-2 establishes rules between sets of parametric values of morphs describing open variable behavior and individual values of hidden variables.

4.3.3 The GPR-3 pools rules that connect the same open variable and hidden variable and satisfy certain statistical and rule-generation criteria. The number and credibility of rules increase with experience.

4.3.4 The GPR-4 estimates the values of hidden variables at desired time points.

4.3.5 The GPR-5 extends the system to distributed processing and intelligence. It merges source files and knowledge bases, established by satellite computers at different observation points, if certain statistical and file-generation criteria are satisfied--as verified by the system automatically.

4.3.6 The GPR-6 extends the system's capabilities to estimating the functional form of hidden variable distributions rather than estimating only individual values of hidden variables.

5. ACKNOWLEDGEMENTS

It is impossible to list every past member of the now extinct "Poker Group". We estimate that more than 40 people have, at one time or another, contributed to its efforts and we are grateful for their ideas, programming and critical comments. NSF Grants GJ-658, GJ-658/A*1, and MCS76-2478 provided partial support to these studies.

The experience gained in the work on Draw Poker is utilized in our current activity on automatic analysis and synthesis of strategies. We are indebted to the other members of our Group for Computer Studies of Strategies: (in random order) Shinji Moriya, Ron Lo, Neal Nazur, Bill Chang, Robert Cromp, Paul Duerig, Paul Ehrler, Michael Belofsky, Richard Wozniak, Gerald Donlon, Nancy Strohmeier and Kulbir

Arora. The efforts of this group, as well as writing this paper, have been supported by the NSF Grant #1-8220.

Finally, we acknowledge the permission of the Association for Computing Machinery, Inc. to reprint Figure 6 and excerpts of our prior publication [5].

6. REFERENCES

[1] Findler, N. V., H. Klein, W. Gould, A. Kowal, and J. Menig: Studies on decision making using the game of Poker (Proc. IFIP Congress 71, Ljubljana, Yugoslavia, pp. 1448-1459; North-Holland: Amsterdam, 1972).

[2] Findler, N. V.: Computer experiments on forming and optimizing heuristic rules (In A. Elithorn and M. Jones (Eds.): Artificial and Human Thinking. Elsevier: Amsterdam, 1973).

[3] Findler, N. V., H. Klein, and Z. Levine: Experiments with inductive discovery processes leading to heuristics in a Poker program (In Beckmann, Goos, and Künzi (Eds.): Cognitive Processes and Systems, Springer: Berlin, 1973).

[4] Findler, N. V., H. Klein, R. C. Johnson, A. Kowal, Z. Levine and J. Menig: Heuristic programmers and their gambling machines (Proc. ACM Nat. Conf., San Diego, CA, pp. 28-37, 1974).

[5] Findler, N. V.: Studies in machine cognition using the game of Poker (Comm. ACM, 20, pp. 230-245, 1977).

[6] Findler, N. V.: Computer Poker (Scientific American, pp. 144-151, July 1979).

[7] Findler, N. V. and J. van Leeuwen: The complexity of decision trees, the Quasi-Optimizer, and the power of heuristic rules (Information and Control, 40, pp. 1-19, 1979).

[8] Findler, N. V., G. L. Sicherman, and S. Feuerstein: Teaching strategies to an Advice Taker/Inquirer system (Proc. Euro IFIP79 Conf., London, England, pp. 457-465, 1979).

[9] Findler, N. V.: Aspects of computer learning (Cybernetics and Systems, 11, pp. 65-84, 1980). Also reprinted in N. V. Findler, W. Horn, and R. Trappl (Eds.):

Progress in Cybernetics and Systems Research, Vol. 11, Hemisphere Publishing: Washington DC, 1982).

[10] Findler, N. V.: Pattern recognition and generalized production systems in strategy development (Proc. Fifth Internat. Conf. on Pattern Recognition; Miami, Fla., Vol. I, pp. 140-150, 1980).

[11] Findler, N. V. and J. P. Martins: On automating computer model construction--The second step toward a Quasi-Optimizer system (Journal of Information and Optimization Sciences, 2, pp. 119-136, 1980).

[12] Findler, N. V.: A multi-level learning technique using production systems (Cybernetics and Systems, to appear).

[13] Findler, N. V. and E. Morgado: Morph-fitting--An effective technique of approximation (IEEE Trans. on Pattern Analysis and Machine Intelligence, Special Issue: Computer Intelligence--Three Decades, to appear).

[14] Findler, N. V.: An expert subsystem based on generalized production rules (Submitted for publication).

[15] Findler, N. V., N. Mazur, and B. McCall: A note on computing the asymptotic form of a limited sequence of decision trees (Submitted for publication).

[16] Findler, N. V., J. E. Brown, R. Lo, H. Y. You: A module to estimate numerical values of hidden variables for expert systems (Submitted for publication).

[17] Findler, N. V.: On a system that dynamically generates its own experimental design (Submitted for publication).

[18] Findler, N. V.: Toward a theory of strategies (Submitted for publication).

[19] Findler, N. V.: An overview of the Quasi-Optimizer system (Submitted for publication).

[20] Findler, N. V.: Automatic analysis and synthesis of strategies -- A new branch of Artificial Intelligence (Submitted for publication).

[21] Crawford, J. R.: How To Be a Consistent Winner in the Most Popular Card Games. (Dolphin Books: Garden City, New York, 1961).

[22] Jacoby, O. Oswald Jacoby on Poker. (Doubleday: Garden

City, New York, 1947).

[23] Morehead, A. H. (Ed.): Official Rules of Card Games. (Crest Book: New York, 1964); and Morehead, A. H., Frey, R. L., and Mott-Smith, G.: The New Complete Hoyle. (Garden City Books: Garden City, New York, 1964).

[24] Yardley, H. O.: The Education of a Poker Player. (Pocket Books: New York, 1961).

[25] Zadeh, N.: Winning Poker Systems. (Prentice-Hall: Englewood Cliffs, New Jersey, 1974).

[26] Wallace, F. R.: Advanced Concepts of Poker. (I & O Publishing Co.: Wilmington, Deleware, 1968).

[27] Roy, T. L.: The Statistically Fair Poker Player (Unpublished Master's Project, Department of Computer Science, State University of New York at Buffalo, 1976).

[28] Pearson, C. E.: The Zadeh Poker Strategy and How I Approximate It (Unpublished Master's Project, Department of Computer Science, State University of New York at Buffalo, 1981).

[29] Sicherman, G. L.: Implementing Zadeh's bluffing strategy (Poker Group Internal Memo No. 234, 1979).

APPENDIX

Outline of the Rules of Draw Poker

As Played by the SUNY-Buffalo Poker System

A standard 52-card pack is used. After each game, the turn to deal passes to the left. Before each deal, every player pays a fixed number of chips (the ante) into a pool (the pot) which will ultimately be awarded to the player with the best hand.

The dealer deals five cards face down to every player. The game then passes through six States:

1. Pre-draw opening state. Starting with the player at the dealer's left, each player either opens by announcing a bet and paying the stated amount into the pot, or checks (i.e. passes) by betting nothing. As soon as some player opens, the game enter State 2. If no player opens, the same

player shuffles the cards and deals again.

2. Pre-draw betting state. Every player in turn may fold (i.e. drop) by paying nothing and withdrawing from the game, call by paying enough to make his total contribution equal to the current bet, or raise by increasing the bet and paying enough to make his total contribution equal to the new bet. State 2 ends when each player has either met the current bet (i.e. stayed alive) or folded.

3. Drawing state. Each active player in turn may discard some of his cards face down. (In real games of Poker, it is customary to limit the exchange to three cards.) The dealer gives him the same number of new cards face down from the undealt portion of the pack.

4. Post-draw opening state. This state is just like State 1 except that players who have folded do not take part and if no player opens, the game proceeds to State 6.

5. Post-draw betting state. This state is just like State 2.

6. Showdown state. The players who have not folded reveal their hands in unison. The player with the highest-ranking hand wins the pot. In a tie, the winners share the pot equally.

In descending order of strength (and rarity), the nine types of Poker hands are:

1. Straight Flush: five cards of the same suit and in sequence; e.g., H9-H8-H7-H6-H5.

2. Four of a Kind: four cards of the same rank; e.g., SK-HK-DK-CK-H6.

3. Full House: three cards of one rank and two cards of another; e.g., S2-D2-C2-HJ-CJ.

4. Flush: five cards of the same suit; e.g., DA-D10-D8-D5-D3.

5. Straight: five cards in sequence, regardless of suit; eg., H5-C4-C3-S2-HA.

6. Three of a Kind: three cards of the same rank; e.g., H10-D10-C10-DA-C4.

7. Two Pair: two cards of one rank and two of another; e.g., S9-C9-H5-C5-H8.

8. Pair: two cards of the same rank; e.g., D6-C6-S10-C5-H2.

9. High-card: a hand belonging to none of the above types; e.g., SA-CQ-H9-S8-S4.

The cards rank Ace(highest)-King-Queen-Jack-10-9-8-7-6-5-4-3-2(lowest), except that in straights and straight flushes the Ace may rank high (A-K-Q-J-10) or low (5-4-3-2-A). Hands of the same type are adjudged by the ranks of their cards; for example, K-K-3-3-6 ("kings up") beats Q-Q-J-J-A ("Queens up"), and 9-9-9-3-3 ("nines full") beats 8-8-8-K-K ("eights full").


FILM
7-